Core Grid Functions: A Minimal Architecture for Grids

William E. Johnston

Lawrence Berkeley National Lab and NASA Ames Research Center

(www-itg.lbl.gov/~wej)

Work of the GGF Grid Protocol Architecture WG
W. Johnston, J. Brooke, white paper co-authors

Core Grid Functions

- Goal:
  - A minimal set of Grid functions that provide uniform interfaces and management for architecturally, geographically, and administratively heterogeneous computing, data, and instrument systems
  - that are managed as production Grids
- “Production Grids” are the Grids that are trying to provide services to a diverse user community to whom the operators of the Grid are responsible for providing a reliable and useful service
- Note: Interoperability also requires operational agreements

Core Grid Functions

- This minimal set of functions are the smallest set of services that are needed to build all other Grid frameworks, middleware, and applications
  - the minimal services may vary somewhat depending on the type of Grid resource – computing, data, instrument, etc.

Core Grid Functions

- Defining a “minimal” set of functions is important because:
  - They provide a metric related to whether a system is a Grid enabled system, or not
  - without the Core Grid Functions, there will be Grid middleware, frameworks, and applications that cannot function
  - represent the fundamental persistent infrastructure of the Grid
  - represent most of the operational effort in building and managing Grids

Core Grid Functions

- Criteria for a Core Grid Function
  - cannot be built on top of other Grid services
  - is essential for building other Grid services and applications, or for providing scalability or security
  - must be self contained (except possibly with respect to security)

Portals

Higher Level Services (applications, utilities, etc.)

Frameworks (Legion-G, CORBA, ….)

Toolkits and collective services

- Brokering
- Job mg'mt (e.g. Condor-G, Unicore)
- DataGrid services (e.g. replication and naming)
- Workflow engine

Encapsulation as Python Services, Script Based Services, Java Based Services, …

Core Grid Functions – protocol endpoints

Globus 2-style interface

- Authentication and Security
- Resource discovery
- Resource Scheduling
- Events and Monitoring
- Uniform Computing Access
- Uniform Data Access
- Communication

Configuration based workflow transformation

OGSA

- Service discovery
- Lifecycle management
- Service registry
- Service factory (execution)
- Service handleMap
- Notification (events)

Mg'mt Access (remote shell & cpy)

Grid Auxiliary Functions

Identity Credential Management

Grid Security Infrastructure

Communications

space-based networks | optical networks | Internet | …

Security Gateways

Core Grid Functions – protocol endpoints

Grid Security Infrastructure: Authentication (human, host, service), delegation/proxy, secure communication

Uniform Computing Access

Execution environment establishment | OGSA hosting environment | Unix shell hosting environment

Uniform Data Access

Resource Scheduling

Mg'mt Access (remote shell & cpy)

Authorization

Events, Monitoring, Logging

Grid Auxiliary Functions

Persistent state and Registry

- resource characteristics, internal architecture, operating state, dynamic registry
- event data types
- Dataset replica info.
- VO information

Distributed Resources

scientific instruments 

clusters 

pools of work-stations

tertiary storage 

national super-computer facilities 

Proxy servers (NAT, FTP cache, etc.)

information servers

- J2EE hosting environment servers
- Factory services

Core Grid Functions

- Resource Discovery and State / Grid Persistent State
- Resource Scheduling
- Uniform Computing Access
- Uniform Data Access
- Asynchronous Information Sources
  (Events, Monitoring, Logging, etc.)
- Remote Authentication, Authorization, Delegation, and Secure Communication
- System Management and Access
- Architectural Constraints (e.g. security)
- Bindings

Resource Discovery & State / Grid Persistent State

- A Grid information service must provide information about existence and characteristics of all Grid resources
- Should minimize the number of persistent information servers needed to enable Grid services and applications
- Functionality
  - Provide for locating all Grid resources with specified properties, within a certain scoping
  - Provide state information as pointers
  - Accommodate a dynamic resource base
  - Be extensible to “all” Grid persistent state
    all Grid services can be sources of information, and if this information needs to be referenced and/or discovered, it should be possible to store and/or represent it in the Grid information service. E.g.
    - Data from users, Virtual Organizations, applications
    - Computing resources
    - Available software
    - Current user allocation
    - Asynchronous Information Sources registry and data content

Resource Discovery / Grid Persistent State

- A minimal service
  - Discovery is an essential Grid function. Without discovery, you cannot build virtual systems from dynamically changing pools of resources.
  - Management of persistent servers is operationally expensive, therefore it is critical to minimize the number of servers needed for a persistent Grid
    - Storing / representing all manner of persistent Grid information with one service is important to minimize required operational support

Resource Scheduling

- Scheduling coordinates distinct resources so that they may operate cooperatively
- Functionality
  - Establish a given virtual system relationship among an administratively independent set of Grid resources via co-scheduling
  - Return information sufficient for negotiation of a common QoS (e.g. time slot) among independent resources
  - A scheduler operating on the resource must
    - Provide time of day reservation
    - Evaluate the future availability of a reservation request and pass that information back to the requester
    - Support soft reservations to allow time for an external broker to negotiate a common reservation among several resources
- A minimal service
  - Essential for QoS
  - Not possible to emulate

Uniform Computing Access

- Job / process initiation
- Functionality
  - Initiate a process or task script on a remote Grid system
  - Support queries about queue types
  - Support submission to named queues (different classes of service)
  - Perform access control based on Grid identity
  - Adapt to variations in system architecture

Uniform Computing Access

- Execution environment establishment
  - Hosting (provide for certain Grid styles of I/O, IPC, etc.)
    - OGSA
    - Unix shell
  - Establish the application runtime environment
  - Configuration based workflow transformation

Uniform Data Access

- Today the primary Grid data access is to named, unstructured objects (“flat” files)
  - objects / files whose structure is understood only by the application that reads the files, and not by the storage system
  - Hence, the primary current model for Grid data access is FTP
- Other emerging functionality in Grid storage resources:
  - Support for some mechanism of sub-setting or filtering data before it leaves the storage resource
  - providing access to relational databases
  - providing access to object oriented databases (?)
- Flat File / Unstructured Object Access Functionality
  - Storage access abstraction
  - Partial file access
  - Integrated Grid security infrastructure security and access control based on the Grid identity

Asynchronous Information Sources (Events, Monitoring, Logging, etc.)

- "Asynchronous Information Sources" = any source of XML formatted objects that can publish its existence and object content characteristics, and then support subscription based delivery of those objects
- Functionality
  - Source registration (a la GMA, the source registers its existence and the content of the objects that it will generate)
  - Registry should be "globally" searchable based on various source and/or object content characteristics
  - Receiving data is by subscription and by direct transfer (source to sink) – the GMA model
- A minimal service
  - Generally, users cannot start persistent servers
  - May be required on systems where jobs cannot be initiated (e.g. storage and instrument control systems)

Remote Authentication, Authorization, Delegation, and Secure Communication

- Identity Certification Authority and certificate management
  - Provides a mechanism for users / entities to request certificates
  - Provides a registration process that verifies user/entity identity
  - Issues and signs X.509 identity certificates
  - Provides Certificate Revocation List generation, management, access, and use
  - Provides a certificate repository
  - Has a formal policy

Remote Authentication, Authorization, Delegation, and Secure Communication

- Authentication
  - Authenticate user access based on Grid identity cert
  - Provide for using host identity credentials at both ends of a transport connection for
    - validating the system identities
    - Securely conveying user entity credentials/proxy to the remote system
- Authorization
  - Access control based on Grid identity and attributes
- Secure Communication
  - Encrypted streams and messages
- Delegation
  - The process by which a user's identity (perhaps with restrictions) is carried to a remote system without the user being directly involved at the remote system
- These are all essential components for secure, authenticated, and authorized access to remote systems

System Management and Access

- Remote system management, and sometimes remote user access, are needed so that Grid resources may be managed and interactively accessed within the Grid context
- Functionality
  - Remote login, authenticated and secured with Grid security functions and authorization based on Grid identity
  - Remote shell, authenticated and secured with Grid security functions and authorization based on Grid identity
  - Remote copy, authenticated and secured with Grid security functions and authorization based on Grid identity
- This seems to be an essential service, because if it is not provided then it is always accomplished in a ad-hoc manner

Architectural Constraints

- In order to be called a Grid Service, it should not be possible to convey command and control messages to remote Grid systems except through the secure and authenticated communication provided by the Grid security functions
- Secure data channels should always be optional, as encryption may be impractical in the cases of high data rates or volumes

Bindings

- Most of the Core Functions will be defined in terms of protocols and data structures, and this provides the basic uniformity required of Grids
- However, there will be many ways to use these Core Functions. For example
  - Globus toolkit's C language
  - CoG kit's Java interface to the Globus functions
  - PyGlobus interface to the Globus functions
- Arguably the OGSI work represents a non- "Globus" interface to the Core functions
- And there will be others

- This talk is at grid.lbl.gov/~wej/Grids